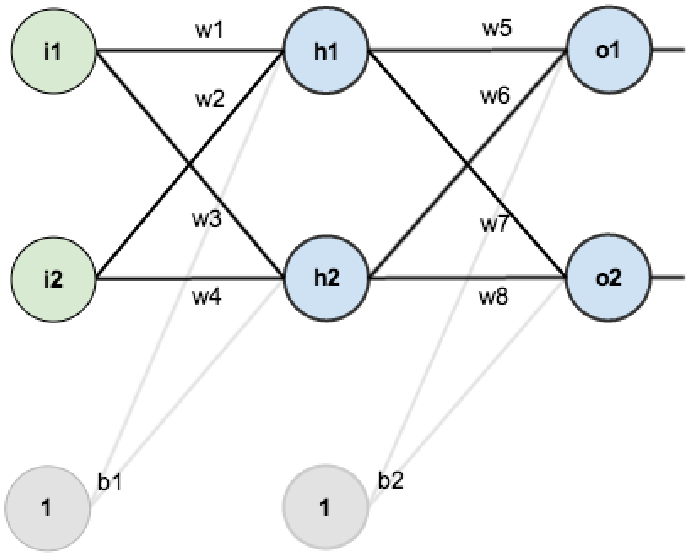
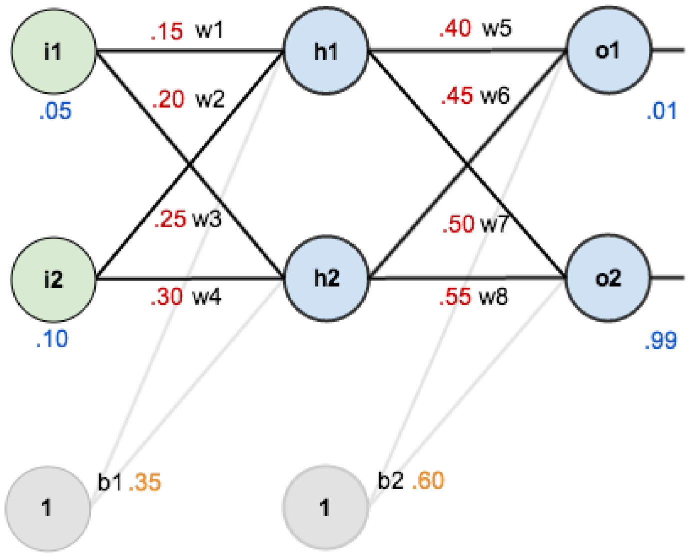
一步一步Backpropagation反向传播示例

对于本教程，我们将使用具有两个输入，两个隐藏神经元，两个输出神经元的神经网络。另外，隐藏和输出的神经元将包括一个偏差bias。

这是基本的结构：



为了有一些数字可以使用，这里有初始权重，偏差和训练输入/输出：



反向传播的目的是优化权重，使得神经网络可以学习如何将任意输入正确地映射到输出。

对于本教程的其余部分，我们将使用单个训练集：给定输入为0.05和0.10，我们希望神经网络输出为0.01和0.99。

向前传播

我们找出每个隐层神经元的总净输入，使用激活函数压缩总净输入（这里我们使用logistic函数），然后用输出层神经元重复该过程。

以下是我们如何计算h1的总净输入：

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

然后我们使用logistic函数来压缩它以获得h1的输出：

$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$

对于h2执行相同的过程，我们得到：

$$out_{h2} = 0.596884378$$

我们对输出层神经元重复这个过程，使用隐藏层神经元的输出作为输入。

以下是o1的输出：

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$net_{o1} = 0.4 * 0.593269992 + 0.45 * 0.596884378 + 0.6 * 1 = 1.105905967$$

$$out_{o1} = \frac{1}{1+e^{-net_{o1}}} = \frac{1}{1+e^{-1.105905967}} = 0.75136507$$

并为o2执行相同的过程，我们得到：

$$out_{o2} = 0.772928465$$

计算总误差

我们现在可以使用平方误差函数计算每个输出神经元的误差
，并将它们相加得出总误差：

$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

加入$\frac{1}{2}$以便在稍后求微分时，消掉指数。结果最后还要乘以学习率，所以我们在这里引入一个常数并不重要。

例如，o1的目标输出为0.01，但神经网络输出为0.75136507，因此其误差为：

$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

重复o2的这个过程（记住目标是0.99），我们得到：

$$E_{o2} = 0.023560026$$

神经网络的总误差是这些误差的总和：

$$E_{total} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371109$$

向后传播

我们的反向传播目标是更新网络中的每个权重，从而使实际输出更接近目标输出，从而最大限度地减少每个输出神经元和整个网络的误差。

Output Layer输出层

思考 w_5 ，我们想知道w5的变化会对总误差产生多大影响 aka $\frac{\partial E_{total}}{\partial w_5}$

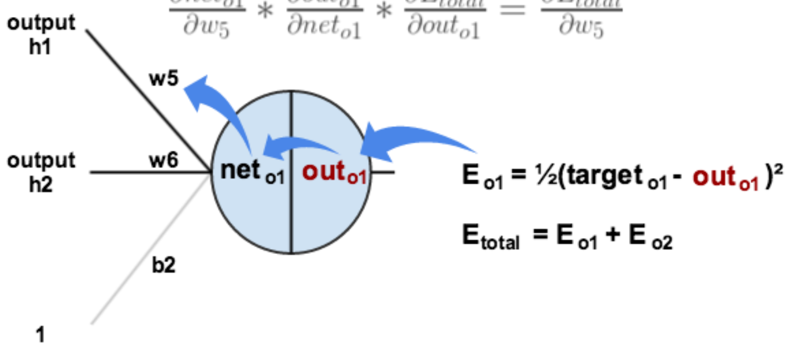
$\frac{\partial E_{total}}{\partial w_5}$读作$E_{total}$ 对 w_5的偏导也可以称作相对于 w_5 的梯度

通过应用链式法则，我们知道：

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

从视觉上来看，我们正在做的是：

$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$



我们需要弄清这个方程式的每个部分。

首先，相对于输出，总误差有多大变化？

$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = 2 * \frac{1}{2}(target_{o1} - out_{o1})^{2-1} * -1 + 0$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = -(target_{o1} - out_{o1}) = -(0.01 - 0.75136507) = 0.74136507$$

$-(target - out)$ 有时表示为 $out - target$

当我们对 out_{o1} 取总误差的偏导数时，$\frac{1}{2}(target_{o2} - out_{o2})^2$ 变为0，因为 out_{o1} 不影响它，常量的导数等于0

接下来，o1的输出对于其总净输入有多少变化？

logistic函数的偏导数是输出乘以（1减去输出）：

$$out_{o1} = \frac{1}{1+e^{-net_{o1}}}$$

$$\frac{\partial out_{o1}}{\partial net_{o1}} = out_{o1}(1 - out_{o1}) = 0.75136507(1 - 0.75136507) = 0.186815602$$

最后，o1的总净输入对w5有多少变化？

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$\frac{\partial net_{o1}}{\partial w_5} = 1 * out_{h1} * w_5^{(1-1)} + 0 + 0 = out_{h1} = 0.593269992$$

把它们放在一起：

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

你会经常以delta规则的形式将此计算结合在一起：

$$\frac{\partial E_{total}}{\partial w_5} = -(target_{o1} - out_{o1}) * out_{o1}(1 - out_{o1}) * out_{h1}$$

或者，我们可以将 $\frac{\partial E_{total}}{\partial out_{o1}}$ 和 $\frac{\partial out_{o1}}{\partial net_{o1}}$ 写成 $\frac{\partial E_{total}}{\partial net_{o1}}$

aka δ_{o1}（希腊字母delta）也称为节点增量。我们可以用这个来重写上面的计算：

$$\delta_{o1} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} = \frac{\partial E_{total}}{\partial net_{o1}}$$

$$\delta_{o1} = -(target_{o1} - out_{o1}) * out_{o1}(1 - out_{o1})$$

因此：

$$\frac{\partial E_{total}}{\partial w_5} = \delta_{o1} out_{h1}$$

一些来源从 δ 提取负号，因此它将被写为：

$$\frac{\partial E_{total}}{\partial w_5} = -\delta_{o1} out_{h1}$$

为了减少误差，我们从当前权重中减去该值（可选的乘以一些学习速率eta，我们设置为0.5）：

$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

一些来源使用α (alpha)来表示学习率，也有使用η (eta) 的，甚至还有使用ϵ (epsilon) 的

我们可以重复这个过程来获得新的权重 w_6, w_7和 w_8：

$$w_6^+ = 0.408666186$$

$$w_7^+ = 0.511301270$$

$$w_8^+ = 0.561370121$$

在我们拥有进入隐藏层神经元的新权重之后，我们在神经网络中执行实际的更新（即，当我们继续下面的反向传播算法时，我们使用原始权重而不是更新的权重）。

Hidden Layer隐藏层

接下来，我们将通过计算 w_1, w_2, w_3和 w_4 的新值来继续向后传播

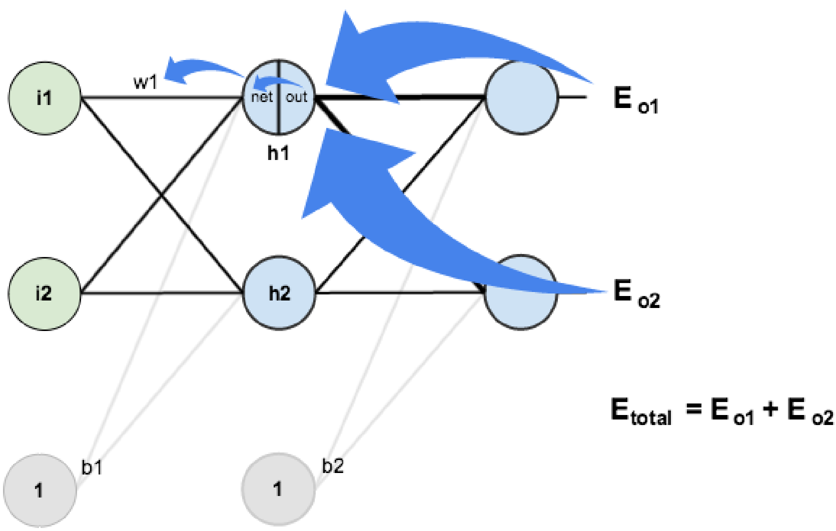
这是我们需要弄清楚的：

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

从视觉上来看：

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



我们将使用与输出层类似的过程，但略有不同，因为每个隐藏层神经元的输出对多个输出神经元的输出（因此也是错误）有贡献。我们知道out_{h1}会影响out_{o1}和out_{o2}，因此 $\frac{\partial E_{total}}{\partial out_{h1}}$ 需要考虑其对两个输出神经元的影响：

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$

从 $\frac{\partial E_{o1}}{\partial out_{h1}}$ 开始：

$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial out_{h1}}$$

我们可以使用我们之前计算的值来计算$\frac{\partial E_{o1}}{\partial net_{o1}}$：

$$\frac{\partial E_{o1}}{\partial net_{o1}} = \frac{\partial E_{o1}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} = 0.74136507 * 0.186815602 = 0.138498562$$

而$\frac{\partial net_{o1}}{\partial out_{h1}}$等于$w_5$ ：

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$\frac{\partial net_{o1}}{\partial out_{h1}} = w_5 = 0.40$$

将其代入：

$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial out_{h1}} = 0.138498562 * 0.40 = 0.055399425$$

遵循$\frac{\partial E_{o2}}{\partial out_{h1}}$的相同过程，我们得到：

$$\frac{\partial E_{o2}}{\partial out_{h1}} = -0.019049119$$

因此：

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}} = 0.055399425 + -0.019049119 = 0.036350306$$

现在我们有 $\frac{\partial E_{total}}{\partial out_{h1}}$，我们需要找出 $\frac{\partial out_{h1}}{\partial net_{h1}}$ 然后$\frac{\partial net_{h1}}{\partial w}$每个权重 ：

$$out_{h1} = \frac{1}{1+e^{-net_{h1}}}$$

$$\frac{\partial out_{h1}}{\partial net_{h1}} = out_{h1}(1 - out_{h1}) = 0.59326999(1 - 0.59326999) = 0.241300709$$

我们计算总净输入 h_1 相对于 w_1 的偏导数就像我们对输出神经元做的一样：

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$
$$\frac{\partial net_{h1}}{\partial w_1} = i_1 = 0.05$$

把它们放在一起：

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial w_1} = 0.036350306 * 0.241300709 * 0.05 = 0.000438568$$

你也可能会看到写成这样：

$$\frac{\partial E_{total}}{\partial w_1} = (\sum_o \frac{\partial E_{total}}{\partial out_o} * \frac{\partial out_o}{\partial net_o} * \frac{\partial net_o}{\partial out_{h1}}) * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial w_1} = (\sum_o \delta_o * w_{ho}) * out_{h1}(1 - out_{h1}) * i_1$$

$$\frac{\partial E_{total}}{\partial w_1} = \delta_{h1} i_1$$

我们现在可以更新w_1：

$$w_1^+ = w_1 - \eta * \frac{\partial E_{total}}{\partial w_1} = 0.15 - 0.5 * 0.000438568 = 0.149780716$$

对w_2, w_3和w_4进行重复操作：

$$w_2^+ = 0.19956143$$

$$w_3^+ = 0.24975114$$

$$w_4^+ = 0.29950229$$

最后，我们更新了所有的权重！当我们前馈0.05的输入和0.1的输入时，网络上的错误为0.298371109。在第一轮反向传播之后，总错误现在下降到0.291027924。可能看起来不是很多，但是在重复这个过程10,000次后，错误会下降到0.000035085。此时，当我们再次输入0.05和0.1时，两个输出神经元产生0.015912196（vs 0.01目标）和0.98406573（vs目标0.99）。

翻译byGary

原文链接：https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/